PROJECT 3: FUNCTIONAL CALCULATOR USING REMOTE METHOD INVOCATION

RMI stands for Remote Method Invocation.

The following project has been developed using RMI.

RMI allows the clients to invoke methods of a remote interface on remote objects.

RMI does not require any connection protocol like one in CORBA.

The project has been implemented in the following steps:

STEP1:

The project starts with defining an INTERFACE which contains the required methods, parameters, events, arguments, exceptions etc. In order to make the interface REMOTE your interface must extend java.rmi.Remote. The file is saved as “Calculator.java”

STEP2:

Implement the Remote Interface.

- This class CalculatorImpl writes the definitions of all methods defined in the remote interface.
- The class must extend UnicastRemoteObject to make the objects remotely available.

STEP3:

Implement the Server.

The class CalculatorServer contains the main().

Server Class performs the following functionalities:

- The server registers its remote objects with the RMIRegistry.
- It passes the URL and the object reference.
- The first argument is the URL. The URL contains the IP address, default port number and the name of Remote Object.
- The second argument is the remote object reference.
- Calls the constructor from the main().

STEP4:

Implement the Client.

Client class contains the main() and performs the following functionalities:

- Calls the lookup() of the Naming class.
- Passes the URL to identify the reference to the remote object by name.
- Calls the methods of the remote interface on Client requests.
- Fetches the results back to the Clients.

STEP5:

This part describes how I actually made a working calculator.

- Accept a string from the user in the form “+_num1_ num2”.
- Separate the operator and the two operands using substring().
- Using charAt(), store them in three different characters.
- Convert the operator into its ASCII value and the operators into their integer values using Character.getNumericValue().
- Pass the ASCII value of the operator and the two operands to the calculate().
- Store the returned result in a variable and display the result.
- If the client wants to continue then accept another string else invoke the exit().

STEP6:

Running the Calculator Application.

- Compile the interface using the following command: javac Calculator.java
- Compile the Server implementation using the following command: javac CalculatorImpl.java
- Compile the Server with main() using the following command: javac CalculatorServer.java
- Compile the Client using the following command: javac CalculatorClient.java
- Generate the stub and the skeleton using the following command: rmic CalculatorImpl
- Now open three different consoles to run the application.
- On the first console, start the RMI Registry using the following command: rmiregistry
- On the second console, start the Server using the following command: java CalculatorServer
- On the third console, start the Client using the following command: java CalculatorClient.